In [71]:

```
import matplotlib.pyplot as plt
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from pylab import *
```

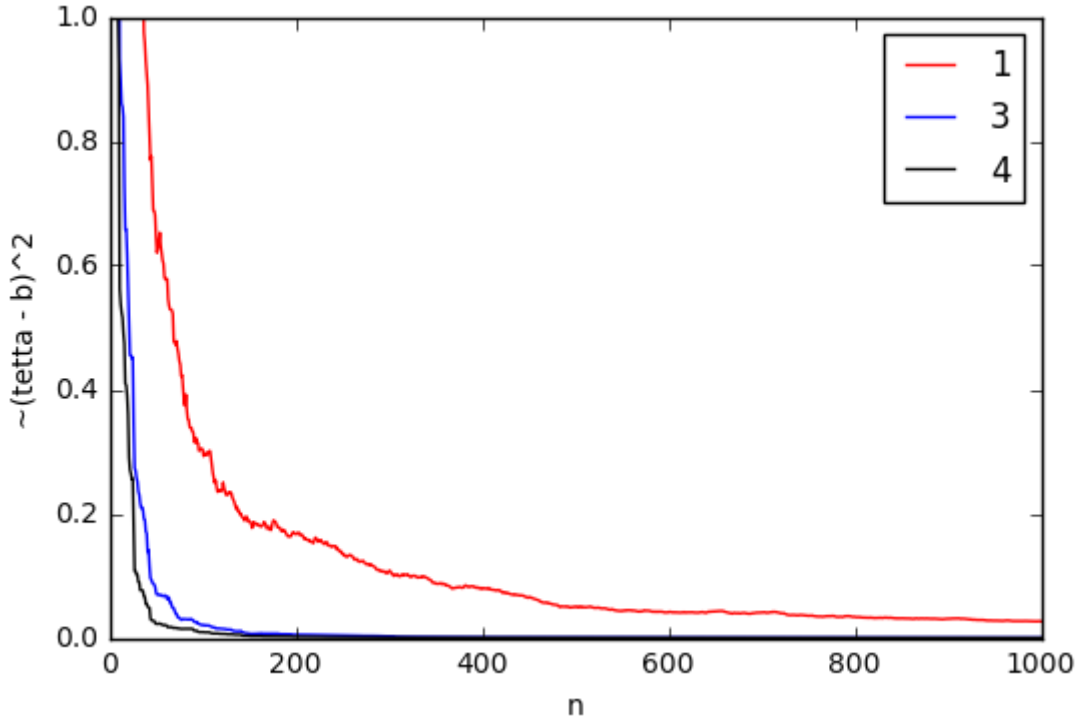
b = 10

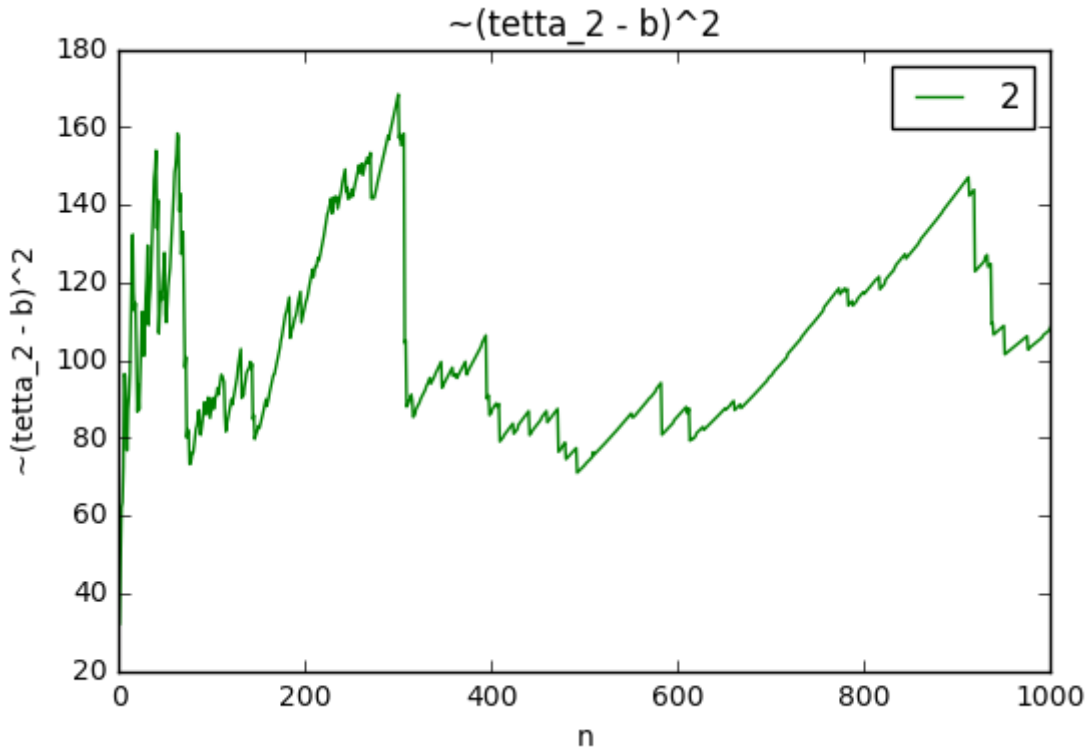
In [72]:

```
# Размер выборки
N = 1000
# Параметр
b = 10
# Массивы для усредненных значений ошибки
arr_1 = np.zeros(1001)
arr_2 = np.zeros(1001)
arr_3 = np.zeros(1001)
arr_4 = np.zeros(1001)
# 100 выборок размера N
for i in range(100):
    s = np.random.uniform(0,b,N)
    for n in range(1, N + 1):
        tetta_1 = 2.0 * s[:n].sum() / len(s[:n])
        tetta_2 = (len(s[:n]) + 1) * s[:n].min()
        tetta_3 = s[:n].min() + s[:n].max()
        tetta_4 = (len(s[:n]) + 1.0) / len(s[:n]) * s[:n].max()
        arr_1[n] = (arr_1[n] + (tetta_1 - b)**2)
        arr_2[n] = (arr_2[n] + (tetta_2 - b)**2)
        arr_3[n] = (arr_3[n] + (tetta_3 - b)**2)
        arr_4[n] = (arr_4[n] + (tetta_4 - b)**2)
# Усреднение по выборкам для каждого n <= N
for i in range(1, N + 1):
    arr_1[i] = arr_1[i] / 100.0
    arr_2[i] = arr_2[i] / 100.0
    arr_3[i] = arr_3[i] / 100.0
    arr_4[i] = arr_4[i] / 100.0
```

In [73]:

```
# Построение графиков. График для оценки 2 построен отдельно.
ylim(0, 1)
plt.plot(range(1, N + 1), arr_1[1:], color = 'red', label='1')
plt.legend(loc='best')
plt.xlabel('n')
plt.ylabel('~(tetta - b)^2');

plt.plot(range(1, N + 1), arr_3[1:], color = 'b', label='3')
plt.legend(loc='best')
plt.xlabel('n')

plt.plot(range(1, N + 1), arr_4[1:], color = 'black', label='4')
plt.legend(loc='best')
plt.xlabel('n')


plt.show()

plt.plot(range(1, N + 1), arr_2[1:], color = 'g', label='2')
plt.legend(loc='best')
plt.title('~(tetta_2 - b)^2')
plt.xlabel('n')
plt.ylabel('~(tetta_2 - b)^2');
plt.show()
```





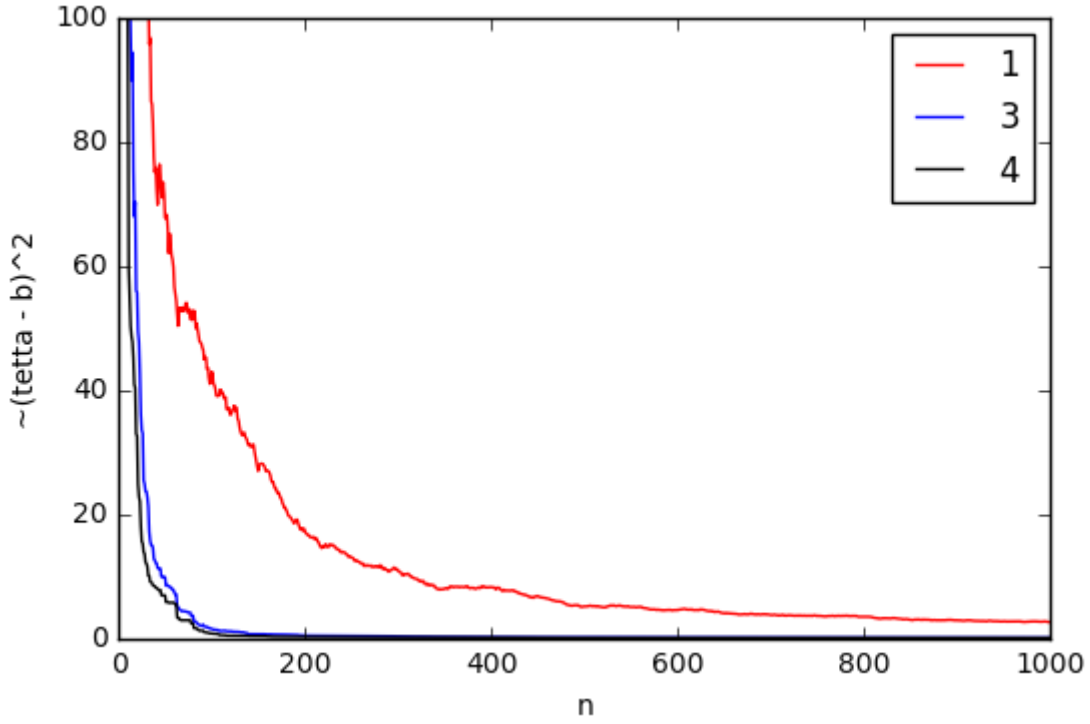
b = 100

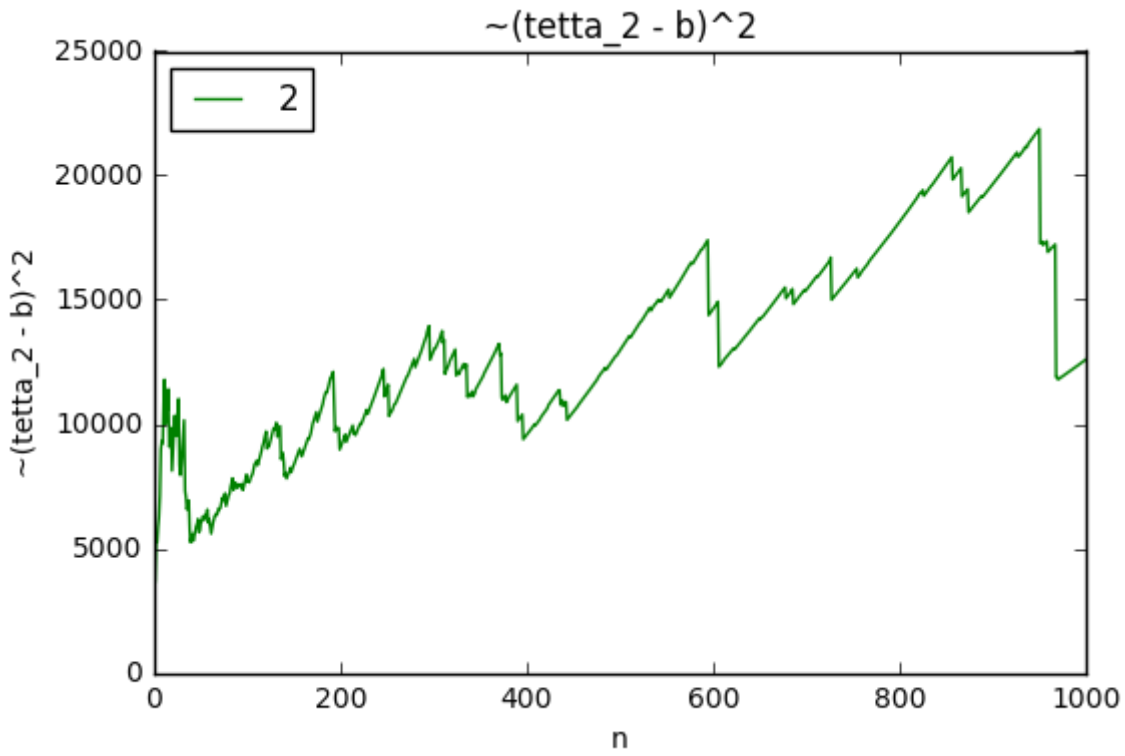
In [74]:

```
# Размер выборки
N = 1000
# Параметр
b = 100
# Массивы для усредненных значений ошибки
arr_1 = np.zeros(1001)
arr_2 = np.zeros(1001)
arr_3 = np.zeros(1001)
arr_4 = np.zeros(1001)
# 100 выборок размера N
for i in range(100):
    s = np.random.uniform(0,b,N)
    for n in range(1, N + 1):
        tetta_1 = 2.0 * s[:n].sum() / len(s[:n])
        tetta_2 = (len(s[:n]) + 1) * s[:n].min()
        tetta_3 = s[:n].min() + s[:n].max()
        tetta_4 = (len(s[:n]) + 1.0) / len(s[:n]) * s[:n].max()
        arr_1[n] = (arr_1[n] + (tetta_1 - b)**2)
        arr_2[n] = (arr_2[n] + (tetta_2 - b)**2)
        arr_3[n] = (arr_3[n] + (tetta_3 - b)**2)
        arr_4[n] = (arr_4[n] + (tetta_4 - b)**2)
# Усреднение по выборкам для каждого n <= N
for i in range(1, N + 1):
    arr_1[i] = arr_1[i] / 100.0
    arr_2[i] = arr_2[i] / 100.0
    arr_3[i] = arr_3[i] / 100.0
    arr_4[i] = arr_4[i] / 100.0
```

In [75]:

```
# Построение графиков. График для оценки 2 построен отдельно.
ylim(0, 100)
plt.plot(range(1, N + 1), arr_1[1:], color = 'red', label='1')
plt.legend(loc='best')
plt.xlabel('n')
plt.ylabel('~(tetta - b)^2');

plt.plot(range(1, N + 1), arr_3[1:], color = 'b', label='3')
plt.legend(loc='best')
plt.xlabel('n')

plt.plot(range(1, N + 1), arr_4[1:], color = 'black', label='4')
plt.legend(loc='best')
plt.xlabel('n')


plt.show()

plt.plot(range(1, N + 1), arr_2[1:], color = 'g', label='2')
plt.legend(loc='best')
plt.title('~(tetta_2 - b)^2')
plt.xlabel('n')
plt.ylabel('~(tetta_2 - b)^2');
plt.show()
```





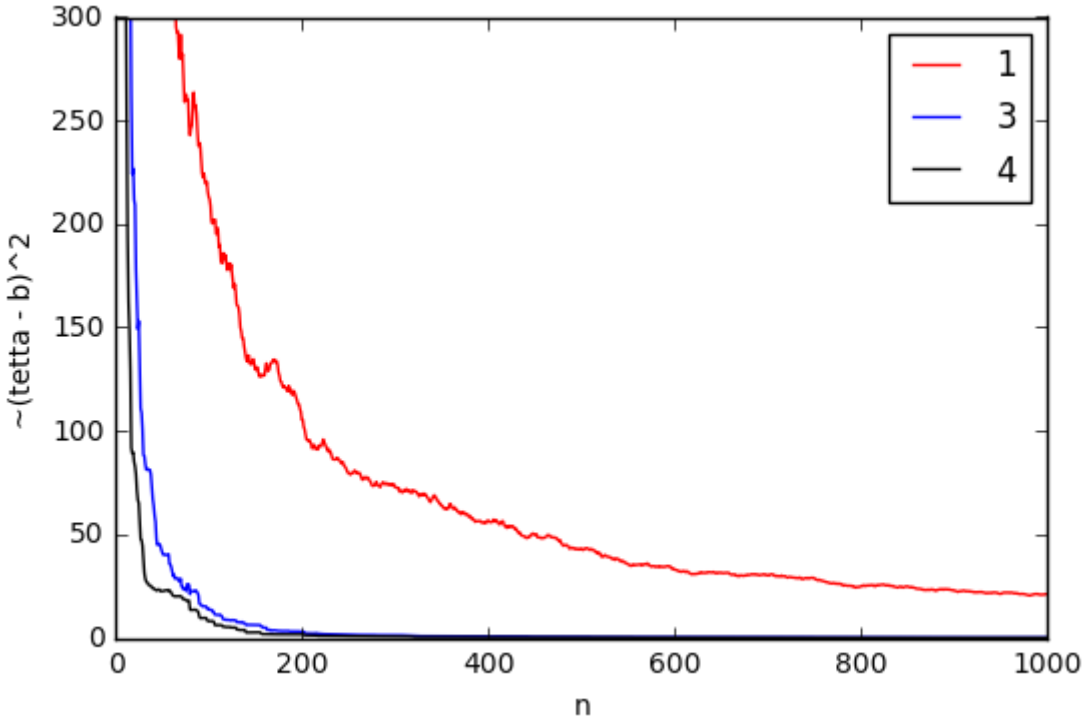
b = 250

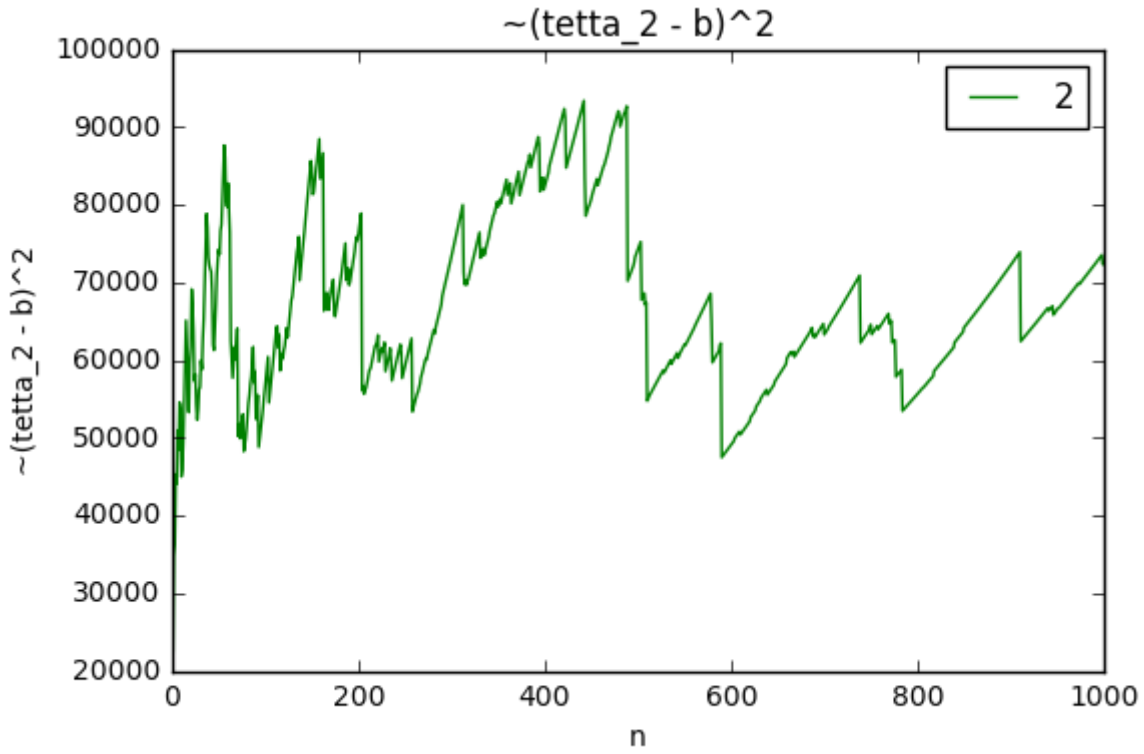
In [76]:

```
# Размер выборки
N = 1000
# Параметр
b = 250
# Массивы для усредненных значений ошибки
arr_1 = np.zeros(1001)
arr_2 = np.zeros(1001)
arr_3 = np.zeros(1001)
arr_4 = np.zeros(1001)
# 100 выборок размера N
for i in range(100):
    s = np.random.uniform(0,b,N)
    for n in range(1, N + 1):
        tetta_1 = 2.0 * s[:n].sum() / len(s[:n])
        tetta_2 = (len(s[:n]) + 1) * s[:n].min()
        tetta_3 = s[:n].min() + s[:n].max()
        tetta_4 = (len(s[:n]) + 1.0) / len(s[:n]) * s[:n].max()
        arr_1[n] = (arr_1[n] + (tetta_1 - b)**2)
        arr_2[n] = (arr_2[n] + (tetta_2 - b)**2)
        arr_3[n] = (arr_3[n] + (tetta_3 - b)**2)
        arr_4[n] = (arr_4[n] + (tetta_4 - b)**2)
# Усреднение по выборкам для каждого n <= N
for i in range(1, N + 1):
    arr_1[i] = arr_1[i] / 100.0
    arr_2[i] = arr_2[i] / 100.0
    arr_3[i] = arr_3[i] / 100.0
    arr_4[i] = arr_4[i] / 100.0
```

In [77]:

```
# Построение графиков. График для оценки 2 построен отдельно.
ylim(0, 300)
plt.plot(range(1, N + 1), arr_1[1:], color = 'red', label='1')
plt.legend(loc='best')
plt.xlabel('n')
plt.ylabel('~(tetta - b)^2');

plt.plot(range(1, N + 1), arr_3[1:], color = 'b', label='3')
plt.legend(loc='best')
plt.xlabel('n')

plt.plot(range(1, N + 1), arr_4[1:], color = 'black', label='4')
plt.legend(loc='best')
plt.xlabel('n')


plt.show()

plt.plot(range(1, N + 1), arr_2[1:], color = 'g', label='2')
plt.legend(loc='best')
plt.title('~(tetta_2 - b)^2')
plt.xlabel('n')
plt.ylabel('~(tetta_2 - b)^2');
plt.show()
```





Вывод

Функции потерь для 1, 3, 4 оценок ведут себя схожим образом, причем для 4 оценки она меньше всего. Функция потерь для 2 оценки сильно отличается, и по значению намного больше.